NASA Technical Memorandum 110335

Experimental Robot Position Sensor Fault Tolerance Using Accelerometers and Joint Torque Sensors

Hal A. Aldridge and Jer-Nan Juang
Langley Research Center, Hampton, Virginia

March 1997

National Aeronautics and
Space Administration
Langley Research Center
Hampton, Virginia 23681-0001

Experimental Robot Position Sensor Fault Tolerance Using Accelerometers and Joint Torque Sensors

Hal A. Aldridge[1] and Jer-Nan Juang[2]
NASA Langley Research Center
Hampton, VA 23681-0001

Abstract

Robot systems in critical applications, such as those in space and nuclear environments, must be able to operate during component failure to complete important tasks. One failure mode that has received little attention is the failure of joint position sensors. Current fault tolerant designs require the addition of directly redundant position sensors which can affect joint design. The proposed method uses joint torque sensors found in most existing advanced robot designs along with easily locatable, lightweight accelerometers to provide a joint position sensor fault recovery mode. This mode uses the torque sensors along with a virtual passive control law for stability and accelerometers for joint position information. Two methods for conversion from Cartesian acceleration to joint position based on robot kinematics, not integration, are presented. The fault tolerant control method was tested on several joints of a laboratory robot. The controllers performed well with noisy, biased data and a model with uncertain parameters.

I. Introduction

Fault tolerant control in robot systems has been an active research area in the past few years[1,2,3,4]. The research has been driven by the need for robots to work reliably in space and nuclear environments where human intervention is difficult. Joint position sensor fault tolerance is an important characteristic of reliable robot systems that has received little attention to date.

For a robot joint to be truly fault tolerant to position sensor failure, it must be driven by a controller which does not rely on position sensor feedback for stability. Control of robotic systems has been a difficult problem due to the nonlinearity of the system equations without considering the complexity added by fault tolerance requirements. Techniques to control this nonlinear system, such as feedback linearization[5], have been popular in the robotic control literature for some time. The main reason for the nonlinearity in the equations is the need to calculate the dynamic effects on the structure. This calculation requires position feedback in addition to a complex system model. Controlling the nonlinear plant based on the full dynamic equations is not the only method for controlling a robot manipulator. If the joints have a torque sensor along the drive axis, the problem can be reduced to controlling the individual joint dynamics. Work in this area has been published recently[6,7]. These methods use known, mostly linear, electric motor driven joint models to control joint acceleration and velocity. However, these methods still require measurement of joint position and velocity to compute the control inputs. A new controller [8] was developed using virtual passive design techniques, joint torque sensor feedback, and simple joint model which is Lyaponov stable without joint position sensor feedback.

Although a controller that is stable without position feedback is available, position feedback is still required to servo to a position. The primary method used in reliable robot design is the inclusion of redundant position sensors. The redundant sensor's output is compared with the primary sensor's output to detect failures and, during failure, is used instead of the primary sensor's output. The addition of the redundant sensor usually affects joint design. Some other fault tolerant design techniques use end effector tracking or kinematic redundancy to continue operation after a sensor failure. The success of these techniques is task and/or environment dependent. A method that is less task dependent and is integrable into existing robot designs would be desirable.

Utilizing double integration, perfect accelerometers would be ideal for this purpose. Current accelerometer designs are small, require little power, and are easily incorporated into existing robot designs. Unfortunately, sensor effects such as noise and bias make the double integration based technique impractical. A method to obtain joint position from Cartesian accelerometers without integration [9] was recently developed to solve this problem. While effective, the number of accelerometers required by this method to provide fault tolerance to all robot joints can be impractical. A system wide method was developed in [10] to reduce the number for accelerometers required.

This paper will review both the torque sensor based joint controller and the joint specific and system wide accelerometer based position determination techniques. The incorporation of these techniques into a fault tolerant control systems will be discussed. The new control systems were implemented on a laboratory manipulator. The results of experiments conducted to

[1] Electrical Engineer, System Integration Branch
[2] Principal Scientist, Structural Dynamics Branch

show the performance of the resulting control systems will be presented.

II. Virtual passive joint control

The virtual passive joint torque control technique is based on a combination of the torque compensation method from Kosuge[5] and the virtual passive control design technique from Juang[11]. The controller design is detailed in Aldridge[8]. Figure 1 shows an electric motor driven, direct drive joint for which the controller will be designed. Equation (1) describes the resulting system.

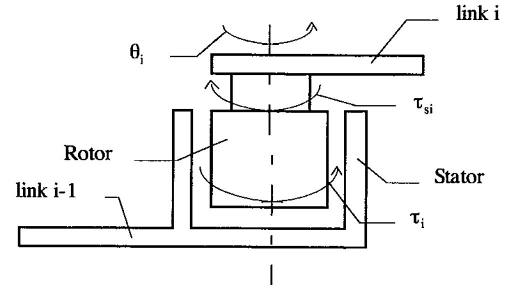

Figure 1: Diagram of proposed direct drive joint

$$\tau_i = m_i\ddot{\theta}_i + \tau_{si} + v_i\dot{\theta}_i + f_i \tag{1}$$

In Eq. (1), m_i is the rotor inertia, v_i is the viscous friction, τ_i is the input torque, τ_{si} is the sensed torque, and f_i is the nonlinear dynamic term resulting from moving the previous i-1 links. Note that all terms relating to link and load dynamic and static torques are grouped into the τ_{si} term. As a result, these torques are sensed, not calculated. For reasonable robot motions, the f_i term is small, is zero for the first link assuming a fixed base, and decreases with system kinetic energy. As a result, the f_i term can be treated as a disturbance and ignored. Equation (2) defines a new quantity, τ_{xi}, ignoring the f_i term.

$$\tau_{xi} = -m_i\ddot{\theta}_i - v_i\dot{\theta}_i + u_i \tag{2}$$

The controller design technique used is similar to the virtual passive control technique presented in Juang. This technique is based on the concept that a mechanical system can be represented by a second-order system with inertia, damping, and stiffness related parameters. An active feedback controller can be designed with its dynamics equivalent to a mechanical system. The resulting controller is,

$$H_M\ddot{x}_c + H_D\dot{x}_c + H_Kx_c + g(y_s) = u \tag{3}$$

where y_s is the measured system output, g is a user defined function, x_c is the controller state vector of dimension n_c, and H_M, H_D, and H_K are the controller mass, damping, and stiffness matrices respectively. These matrices are design parameters and can be chosen to meet performance and stability requirements. The function g is an arbitrary function of the measured system outputs, y_s. These outputs can be system states or combinations of system states.

The torque controller using τ_{xi} will now be detailed. Note that if f_i is treated as a disturbance, $\tau_{xi}=\tau_{si}$. Let:

$$M_{rz} = diag(m_1,...,m_p) \tag{4}$$

$$V = diag(v_1,...,v_p) \tag{5}$$

$$\tau_x = diag(\tau_{x1},...,\tau_{xp}) \tag{6}$$

$$u_i = \tau_i \tag{7}$$

Using the virtual passive controller design technique, a controller that satisfies the Lyaponov stability criteria is given by,

$$\begin{bmatrix} \dot{x}_c \\ \ddot{x}_c \end{bmatrix} = \begin{bmatrix} 0 & I \\ -(R_\tau M_{rz})^{-1}K_c & -(R_\tau M_{rz})^{-1}D_c \end{bmatrix} \begin{bmatrix} x_c \\ \dot{x}_c \end{bmatrix} + \begin{bmatrix} 0 & 0 \\ M_{rz}^{-1} & -M_{rz}^{-1} \end{bmatrix} \begin{bmatrix} \tau_x \\ u' \end{bmatrix} \tag{8}$$

$$u = \begin{bmatrix} K_c & R_\tau V + D_c \end{bmatrix} \begin{bmatrix} x_c \\ \dot{x}_c \end{bmatrix} \tag{9}$$

where D_c, R_τ, and K_c are design matrices. The restriction on these design matrices is that D_c and R_τ must be a symmetric and positive-definite. The previous joint torque command, u', is known and used to calculate the current torque command. Note that the terms x_c and $\dot{x}_c$ are not the joint position and velocity. They are controller states used to satisfy the stability condition. As a result, this controller design can dissipate joint kinetic energy without joint position or velocity feedback.

The resulting controller has several important properties for the fault tolerance application:

- It does not depend on joint position or velocity feedback for stability
- It has tunable stiffness and damping terms
- It can act as an active joint brake by dissipating energy
- Its stability is robust to modeling errors
- It compensates for link and load dynamic effects allowing for reduced position controller gains

III. Joint specific accelerometer based position sensor fault tolerance

Although the virtual passive joint controller does not require position feedback for stability, position feedback is required for position control. In this paper that feedback will be provided by Cartesian accelerometers. The position determination method proposed, detailed in Aldridge[9], requires two triaxial Cartesian accelerometers to be attached to the link driven by the joint whose position sensor is to be made fault tolerant. The method also uses gravity or a known acceleration field as a reference. The accelerometers must be of the type that can sense constant acceleration for acceptable steady state performance. Figure 2 shows a diagram of the proposed accelerometer configuration.

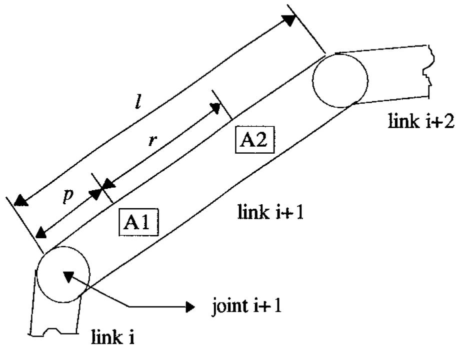

Figure 2: Proposed accelerometer configuration

The problem is to find a measurement which is directly related to joint position. If the acceleration of a point on link i coincident with joint $i+1$ is known from kinematics and measurements of the first i joint states, in the special case where p is the zero vector the joint position is given in Eq. 10.

$$\theta_{i+1} = \tan^{-1}((\hat{a}_y \cos\alpha_i + \hat{a}_z \sin\alpha_i)A1_x - A1_y\hat{a}_x, \\ (\hat{a}_y \cos\alpha_i + \hat{a}_z \sin\alpha_i)A1_y + A1_x\hat{a}_x) \tag{10}$$

where,

$$a_{i+1}^{i} = \begin{bmatrix} \hat{a}_x \\ \hat{a}_y \\ \hat{a}_z \end{bmatrix}, A1 = \begin{bmatrix} A1_x \\ A1_y \\ A1_z \end{bmatrix}$$

and α_i is a known robot Devanit-Hartenberg parameter.

For any joint later in the kinematic chain to be made fault tolerant, the acceleration a_{i+2}^{i+1} must be calculated. The calculated position could be differentiated and used along with the robot kinematics to calculate the acceleration. Due to the noisy nature of accelerometer related signals, differentiation would be impractical. However, given the joint position and the angular velocity and acceleration of the i^{th} link (calculated from the first i measured joint states and kinematics), the relationship between Cartesian acceleration and joint velocity and acceleration is given by Eq. (11).

$$\tilde{A} = \begin{bmatrix} -2\Omega_z r_x & -r_y & -r_x \\ -2\Omega_z r_y & r_x & -r_y \\ 2(\Omega_x r_x + \Omega_y r_y) & 0 & 0 \end{bmatrix}\Theta \tag{11}$$

where,

$$\Omega = R_i^{i+1}\omega_i^i = \begin{bmatrix} \Omega_x \\ \Omega_y \\ \Omega_z \end{bmatrix} \quad r = \begin{bmatrix} r_x \\ r_y \\ r_z \end{bmatrix} \quad \tilde{A} = \begin{bmatrix} \tilde{A}_x \\ \tilde{A}_y \\ \tilde{A}_z \end{bmatrix} \quad \Theta = \begin{bmatrix} \dot{\theta}_{i+1} \\ \ddot{\theta}_{i+1} \\ \dot{\theta}_{i+1}^2 \end{bmatrix}$$

$$\tilde{A} = A2 - A1 - R_i^{i+1}\omega_i^i \times (R_i^{i+1}\omega_i^i \times r) + R_i^{i+1}\dot{\omega}_i^i \times r$$

with R_i^{i+1} the rotation matrix from the i^{th} frame to the $(i+1)^{th}$ frame and ω_i^i the angular velocity of the i^{th} link in the i^{th} frame. The joint velocity and its square are treated as independent variables to keep the equations linear. Although the rank of the solution is not guaranteed, it can be solved by least squares. A simpler solution method involves solving a specific set of equations derived from Eq. (11) depending on zeros in the first column of the matrix in Eq. (11). If r_x=0 or r_y=0, the solution of the equations can be further simplified because the solution for $\dot{\theta}_{i+1}$ is independent from $\ddot{\theta}_{i+1}$.

To solve Eq.(11), three cases must be considered. Let ε be some small number used to bypass small elements of W that would be zero with perfect measurements. Assuming either r_x or r_y is non-zero, the cases are:

1. $|\Omega_x r_x + \Omega_y r_y| > \varepsilon \Rightarrow$ rank(W)=3
2. $|\Omega_x r_x + \Omega_y r_y| < \varepsilon$ and ($|\Omega_z r_x| > \varepsilon$ or $|\Omega_z r_y| > \varepsilon$) $\Rightarrow$ rank(W)=2, equations are underdetermined
3. $|\Omega_x r_x + \Omega_y r_y| < \varepsilon$ and $|\Omega_z r_x| < \varepsilon$ and $|\Omega_z r_y| < \varepsilon$ $\Rightarrow$ rank(W)=2, unique solution with no dependence on $\dot{\theta}_{i+1}$

Solution techniques for the three cases in the general situation of r_x and $r_y \neq 0$ can be derived. However, if the system is designed such that r_x or r_y is zero, the solution techniques become less complex. The following techniques can be used if r_x=0, similar solutions can be shown for r_y=0. For all cases of r_x=0:

$$\ddot{\theta}_{i+1} = \frac{-\tilde{A}_x}{r_y} \tag{12}$$

Now, calculating $\dot{\theta}_{i+1}$ for each case:

Case 1 for r_x=0

$$\dot{\theta}_{i+1} = \frac{\tilde{A}_z}{2\Omega_y r_y} \tag{13}$$

Case 2 for r_x=0

In this case, there exists a quadratic equation in $\dot{\theta}_{i+1}$. Solving with the standard quadratic solution:

$$\dot{\theta}_{i+1} = \frac{\Omega_z r_y \pm \sqrt{r_y(\Omega_z r_y - \tilde{A}_y)}}{r_y} \tag{14}$$

The proper root is selected by checking for consistency with measured accelerations. In contrast to case 3, there is sufficient information in the measured accelerations to determine the proper root.

Case 3 for r_x=0

The solution for case 3 has no explicit dependence on $\dot{\theta}_{i+1}$. As a result, Eq. (15) must be used.

$$\dot{\theta}_{i+1}^2 = \frac{-\tilde{A}_y}{r_y} \tag{15}$$

The magnitude of $\dot{\theta}_{i+1}$ is calculated via a square root. Determination of the sign is problematic and will be addressed later in this section.

Given the results of this calculation in either the general or special case and known robot kinematics, the solution for a_{i+2}^{i+1} can be found without differentiation.

For a real robot system, the assumption that *p* is zero is not practical. However, if *p* is non-zero, the position solution is no longer independent of the velocity and acceleration of the $(i+1)^{th}$ joint. As presented, the joint velocity and acceleration calculation is always dependent on the joint position. This problem can be solved by introducing a correction term and iterating between the joint position and the joint velocity and acceleration solutions. First, the previous joint values are used to calculate the approximate angular velocity and acceleration of the $(i+1)^{th}$ link as shown in Eq. (16) and (17).

$$(\omega_{i+1}^{i+1})_{approx} = R^{i+1}{}_i\,\omega_i^i + (\dot{\theta}_{i+1})_{approx}\,\hat{z} \tag{16}$$

$$(\dot{\omega}_{i+1}^{i+1})_{approx} = R_i^{i+1}\dot{\omega}_i^i + R_i^{i+1}\omega_i^i \times (\dot{\theta}_{i+1})_{approx}\hat{z} + (\ddot{\theta}_{i+1})_{approx}\hat{z} \tag{17}$$

These values are used in Eqs. (18) and (19) to produce a new term, *A1corr*.

$$Acorr = (\dot{\omega}_{i+1}^{i+1})_{approx} \times p + (\omega_{i+1}^{i+1})_{approx} \times (\omega_{i+1}^{i+1})_{approx} \times p \tag{18}$$

$$A1corr = A1 - Acorr \tag{19}$$

The *A1corr* term is used in the position solution instead of *A1* to obtain $(\theta_{i+1})_{approx}$. The $(\theta_{i+1})_{approx}$ term is then used in R_i^{i+1} to obtain a better solution for the joint acceleration and velocity from the a_{i+2}^{i+1} solution. The process is repeated by calculating a new *Acorr* using the new approximation of the joint velocity and acceleration instead of the value at the previous time.

For a physical robot system, this iteration converges rapidly at reasonable sample rates.

If the equations in (11) satisfy only case 3, then the equation for joint velocity is not solvable. The square of the joint velocity is solvable but determining the direction of the velocity is problematic. A method to determine the sign of the velocity and enhance velocity tracking performance in the presence of accelerometer noise in all cases is to implement a velocity observer. The iterative, general position solution is modeled as a linear system expressed by:

$$\begin{aligned} \dot{x} &= Ax + Bu \\ y &= Cx \end{aligned} \tag{20}$$

with state-space matrices:

$$x = \begin{bmatrix} \tilde{\theta}_{i+1} \\ \dot{\tilde{\theta}}_{i+1} \end{bmatrix},\ u = \begin{bmatrix} \theta_{i+1} \\ \ddot{\theta}_{i+1} \end{bmatrix},\ y = \dot{\theta}_{i+1}$$

$$A = \begin{bmatrix} -K_v & 1 \\ -K_k & 0 \end{bmatrix}\ B = \begin{bmatrix} K_v & 0 \\ K_k & 1 \end{bmatrix}\ C = \begin{bmatrix} 0 & 1 \end{bmatrix}$$

The states are joint position $\tilde{\theta}_{i+1}$ and velocity $\dot{\tilde{\theta}}_{i+1}$, the inputs are calculated joint position θ_{i+1} and acceleration $\ddot{\theta}_{i+1}$ from the iterative solution, and the output is the filtered joint velocity, $\dot{\tilde{\theta}}_{i+1}$. The K_k term is used to compensate for possible calculated acceleration bias due to accelerometer sensor error. Without the K_k term, the system is an unstable double integrator and a bias in the calculated acceleration can lead to unbounded joint velocity. The K_k term can cause oscillations in steady state. The K_v term adds damping and can cancel out the oscillations.

IV. System wide accelerometer based joint position sensor fault tolerance

The system wide solution method is designed to complement the joint specific method. Where the joint specific method gives an almost closed form solution for joint position, the system wide method utilizes an optimization algorithm to give an more flexible solution with fewer accelerometers.

The system wide solution method is designed for an *n* revolute joint, rigid link robot system with *m* triaxial accelerometers mounted at various locations on the robot. A robot of this kinematic configuration is governed by the following kinematic equations:

$$\omega_{i+1}^{i+1} = R^{i+1}{}_i\,\omega_i^i + \dot{\theta}_{i+1}\hat{z} \tag{21}$$

$$\dot{\omega}_{i+1}^{i+1} = R_i^{i+1}\dot{\omega}_i^i + R_i^{i+1}\omega_i^i \times \dot{\theta}_i\hat{z} + \ddot{\theta}_{i+1}\hat{z} \tag{22}$$

$$a_{i+1}^{i+1} = R_i^{i+1}(\dot{\omega}_i^i \times l_i + \omega_i^i \times \omega_i^i \times l_i + a_i^i) \quad (23)$$

$$Am_{i+1}^{i+1} = \dot{\omega}_{i+1}^{i+1} \times pm_{i+1} + \omega_{i+1}^{i+1} \times \omega_{i+1}^{i+1} \times pm_{i+1} + a_{i+1}^{i+1} \quad (24)$$

where

ω_i^i angular velocity of i^{th} link in i^{th} frame

a_i^i acceleration of i^{th} link in i^{th} frame

R_i^{i+1} rotation matrix from i^{th} frame to $(i+1)^{th}$ frame

θ_i position of i^{th} joint

Am_{i+1}^{i+1} acceleration of accelerometer Am_{i+1} in the $(i+1)^{th}$ frame

pm_{i+1} vector offset from joint $i+1$ to accelerometer Am_{i+1}^{i+1}

l_i vector offset from joint i to joint $i+1$ in the frame i

$\hat{z}$ identity z vector $[0\ 0\ 1]^T$

Each accelerometer contains information on the position, velocity, and acceleration of joints preceding the accelerometer in the kinematic chain. End-effector mounted accelerometers contain information about all joints. However, this information is mixed together in a nonlinear fashion. Except for accelerometers attached to the first joint, there is no single acceleration that is, in general, dependent on solely the position, velocity, or acceleration of the i^{th} joint. Components of the acceleration due to certain joints can dominate due to kinematic configuration or trajectory. For certain configurations, the end-effector pose does not uniquely determine the joint trajectory of the robot. As a result, only utilizing end-effector mounted accelerometers is not sufficient for determining joint trajectories. Distributing accelerometers along the arm in addition to end-effector mounted accelerometers can help alleviate these problems by obtaining information at points closer to joints of interest.

The proposed solution method utilizes the distributed Cartesian accelerometers in conjunction with any working joint position sensors to recover a lost position sensor. This is done by calculating a knot point, a point in the trajectory having position, velocity, and acceleration, that will make the measured accelerations match the accelerations determined by Eq. (24). In Eq. (25), the data from the operational joint sensors is used to the extent possible, leaving only terms involving the failed joints,

$$Q(\theta_j, \dot{\theta}_j, \ddot{\theta}_j) = A, \quad j \in b \quad (25)$$

where,

b is the set of failed joints

A is the vector of accelerometer measurements $[\, Am_1^1 \ldots Am_m^m \,]^T$

Q is a set of equations of type (24) corresponding to A

If r is the number of failed joints, then the resulting system has $3m$ nonlinear equations (one equation per axis, three axes per accelerometer) in $3r$ (position, velocity, and acceleration of failed joints) unknowns. If this system is solvable, then the knot points for all r joints can be determined and the joint positions calculated.

A minimum criteria for solution is that there be at least as many equations as unknowns, i.e. $m \geq r$. If this condition is not satisfied, then the minimum error solution will be one using least squares. Due to the complexity of the equations, this condition is optimistic. As r approaches m, the system becomes more ill-conditioned. Choosing trajectories which result in a condition number of the gradient of Q less than some reasonable limit can help system solvability.

The system wide method, like the joint specific method, relies on the existence of a known acceleration field, usually gravity. This field provides a known, constant excitation to the system accelerometers. The solution method can be applied to systems without such a field, but positions will be relative, not absolute, and drift due to small errors is likely. The accelerometers must be of the instrument type that can detect constant accelerations.

The system wide solution method requires a computational technique for solving a system of nonlinear equations. To be practical, the solution must be calculable in real-time. Real time in this sense implies updates at a rate fast enough for stable joint control if this method is used for fault recovery. This requirement limits the available solution techniques. The solution technique must also be robust to sensor noise and bias. The convergence of the technique to a good solution must be predictable. Traditional nonlinear solution techniques can be applied to this problem. In the experimental setup detailed in section VI, steepest descent [12] was used for its computational efficiency. Most computational methods require a gradient for rapid convergence. The gradient required for this method can be computed numerically by finite differences [13] or, more efficiently, by the recursive algorithm shown in Appendix A.

V. Fault tolerant control system

Incorporating an accelerometer based position determination method and the virtual passive based torque controller results in a control system which is capable of servoing to a position without joint position sensor feedback. An example of such a control system using the joint specific method from section III can be found in Fig. 3. A similar version using the system wide method from section IV is shown in Fig. 4.

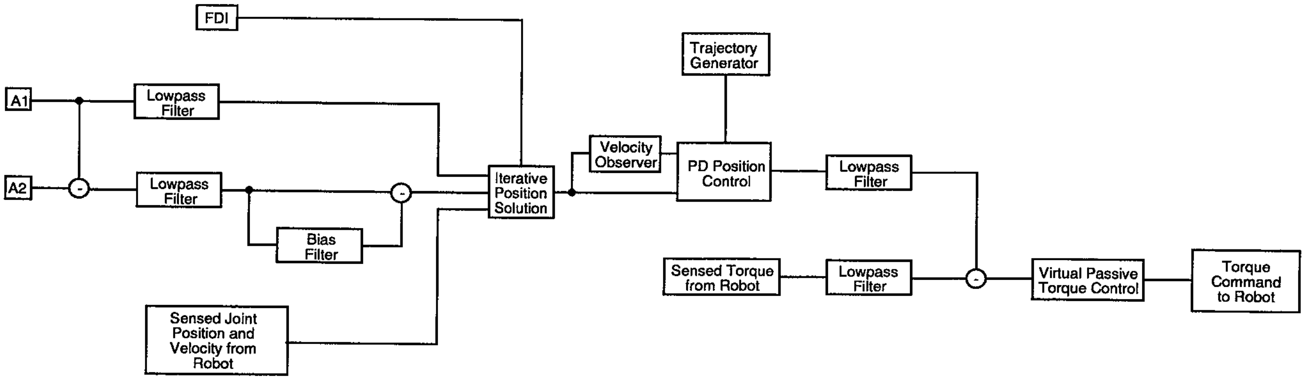

Figure 3: Control system diagram using joint specific method

The FDI, failure detection and identification component, uses existing FDI techniques to determine if a joint position sensor has failed and informs the position solution which joint positions it must calculate. The position solution can be used on unfailed joints as part of the FDI procedure to verify joint position, velocity, and acceleration independently of the joint sensors. For the joint specific method shown in Fig. 3, the output of accelerometer *A1* and the difference between *A2* and *A1*, forming a relative acceleration, are sent to lowpass filters to reduce noise. A bias filter, an extremely lowpass filter, attempts to remove steady state bias in the relative acceleration. The position output of the position solution and the observed velocity are merged with valid joint position and velocity measurements from working joints for the position controller, in this case a simple PD controller. The position command torque and the sensed torque are both filtered to remove noise and match phase shifts. The position command torque is used to offset the sensed torque to drive the robot in the desired direction. The offset sensed toque is processed by the virtual passive torque controller and the commanded torque for the robot joint is calculated.

The control diagram for the system wide method in Fig. 4 is similar with the exception that all system accelerometers are used simultaneously without forming relative accelerations. Although the solution method is capable of producing joint position, velocity, and acceleration, to reduce computational complexity the solution was formulated to only produce position and acceleration. The velocity is the most difficult value to get accurate solutions for in the presence of accelerometer error. The control system used the velocity observer described in section III to produce calculated velocity.

Using the virtual passive torque controller in this manner makes the use of accelerometer based position measurements practical for closed loop control. Since the torque controller compensates for the dynamic and static torques required to drive the links, the position controller gains can be kept low. These low gains are important because, even filtered, the accelerometer based positions are noisier than optical encoder or resolver based measurements. The passive controller gains can also be adjusted to tolerate noisier inputs with the tradeoff of lower position tracking performance. The calculated positions may have a bias associated with them due to sensor error. That position bias would degrade the effectiveness of gravity compensation. By using the passive torque controller, a gravity compensation algorithm is unnecessary.

VI. Experimental setup

Experiments to examine the performance of the proposed method were conducted on a Robotics Research (RRC) 807i manipulator instrumented with

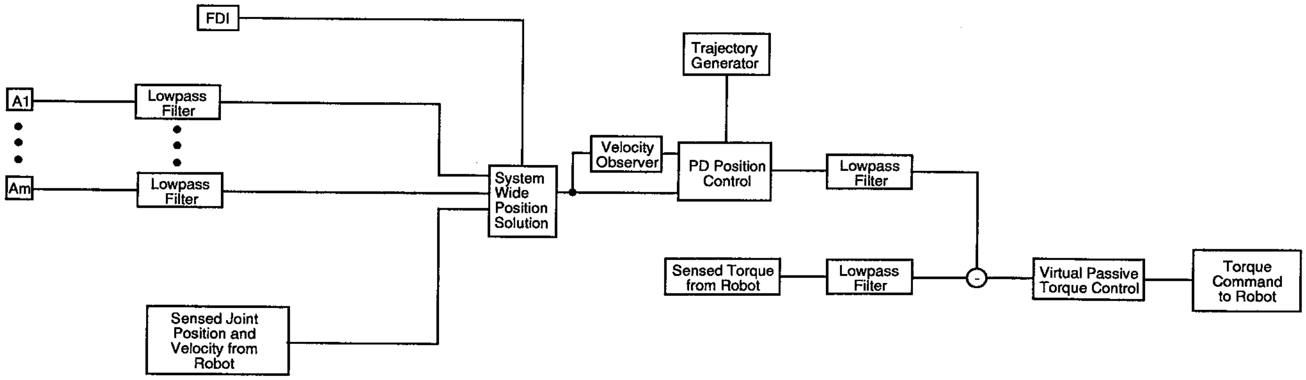

Figure 4: Control system diagram using system wide method

accelerometers. The RRC 807i is a 7 degree of freedom, 0.8m long manipulator with a 10kg payload.

Joint specific setup

The accelerometers for the joint specific method were installed on two joints of the robot in a pattern similar to that in Fig. 2. Figure 5 shows the robot and the accelerometer mounts attached to the shoulder pitch and elbow pitch joints. A closer view of the elbow pitch joint mounted accelerometers is shown in Fig. 6. The accelerometers used were ICSensors model 3140 piezoresistive accelerometers with internal amplification and signal conditioning. The accelerometers had a maximum range of ±2g. These piezoresistive accelerometers were capable of sensing constant accelerations, a necessary feature for the position solution method. Each accelerometer sensed acceleration along a single axis. Three accelerometers were mounted orthogonally to form a single triaxial accelerometer. The accelerometers were chosen for their ease to use and cost. The accelerometer mounts were designed to accommodate the accelerometers and use existing mounting points on the robot. The resulting accelerometer mounts are unsuitable for real world use due to their size and location. However, smaller accelerometers which could be incorporated directly into the robot structure are readily available.

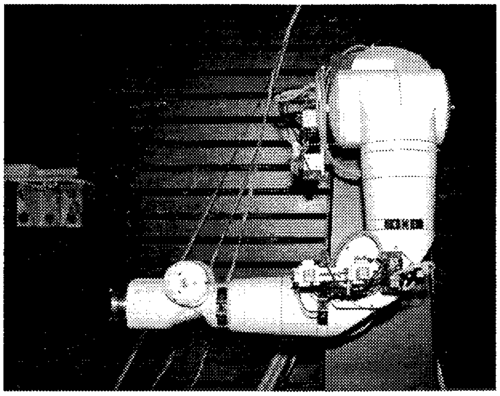

Figure 5: RRC robot with accelerometer mounts

The joint specific based controller was implemented on two 68040 based computers running on a VME backplane. The VME bus was connected to the RRC robot controller via a bus to bus interface. The interface allowed the RRC robot to be controlled by reading and writing to memory locations in the VME address space. The first 68040 based computer, referred to as the control processor, ran the interface code for the arm, arm safety systems, and the virtual passive controller. The control processor generated new torque commands at 200Hz. The second 68040, referred to as the accelerometer processor, communicated with the arm mounted accelerometers and ran the position solution code. The accelerometers were sampled at 100Hz and a new position solution for two concurrent failures was generated at 75Hz.

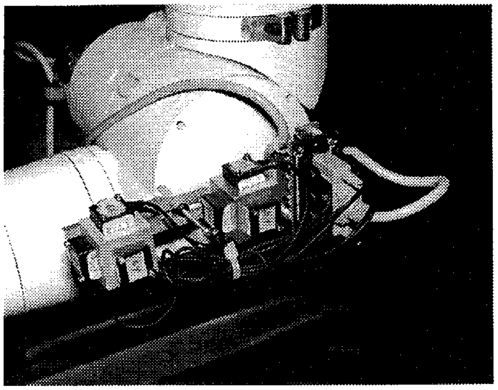

Figure 6: Elbow accelerometer mount

System wide setup

Figures 7 and 8 show the RRC manipulator and the accelerometer mounts for the system wide solution method. Single triaxial accelerometers were attached to the shoulder pitch, elbow roll, and elbow pitch joints, joints 2,3, and 4 respectively. This configuration was the easiest to mount to the robot structure, not the optimal accelerometer configuration. Two triaxial accelerometers were mounted to the end-effector.

A third processor was added to the controller implemented for the joint specific method to handle the increased computational complexity of the system wide method. The third processor, a 33Mhz 68040 called the optimization processor, implemented the system wide position determination algorithm. The final system was capable of producing joint positions for three simultaneous failures at 60Hz. While the optimization processor was fully loaded at this rate, the other processors were not heavily loaded for the system wide experiment.

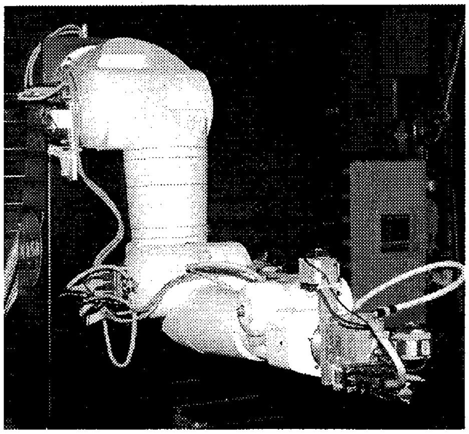

Figure 7: RRC arm with accelerometers, front view

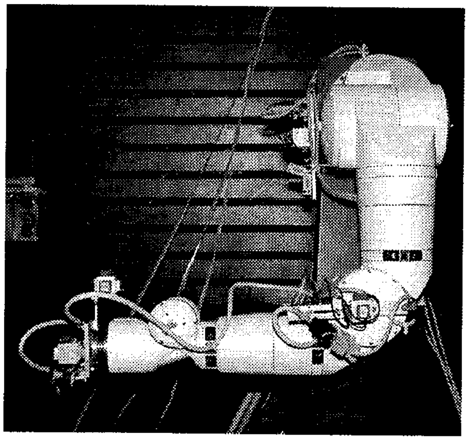

Figure 8: RRC arm with accelerometers, side view

To implement a real-time system for the system wide method on the computational resources available, several compromises were required.

1. Although other solution techniques produce a more robust system, steepest descent optimization was used for its computational efficiency.
2. The gradient was not updated every control cycle to lower computational complexity. A full gradient was available every four cycles. The modularity of the recursive method in Appendix A made distributing the computation over several cycles simple.
3. A new singular value decomposition (SVD) required by the steepest descent method (due to the variable rank of the gradient) was calculated the cycle after a new gradient was completed. The results of the last SVD were used with a current function calculation and the measured acceleration to find the next position solution using the steepest descent algorithm.

Setup notes common to both methods

In instrumenting the robot, electromagnetic noise from the robot and robot controller was a significant problem. The final electrical configuration involved using differential accelerometer outputs and differential A/D conversion. Each side of the differential signal was filtered using a 40Hz analog lowpass filter for noise reduction and anti-aliasing. The bus-to-bus interface contaminated the local ground interfering with the A/D conversion. As a result, the analog lowpass filter was aided by a 5Hz digital lowpass filter to help reduce residual noise. With the existing system noise, the internal accelerometer amplification to 1V/G was essential. If signal quality was improved, less restrictive lowpass filters could be used improving dynamic performance.

The RRC robot was commanded in torque mode. In this mode, the robot controller uses a basic torque controller to overcome joint effects. Its goal is to make the harmonic gear driven joints perform like direct drive joints. The resulting system does reduce geartrain effects but does not eliminate the effects. The virtual passive controller must handle any remaining geartrain friction, hysteresis, or backlash. Although the model presented in section II was direct drive, the controller can be used on gear driven joints if the effects geartrain friction and the gear ratio are considered and nonlinear geartrain effects, such as backlash, are minimized.

The motor parameters required by the controller were not available from the robot manufacturer. Identifying the parameters of motors installed in the robot proved difficult due to the parameters small size and the inability to bypass the low level torque controller. As a result, qualitative data was used to estimate the parameters and the gains chosen to produce the required performance. The resulting performance with substantial modeling errors shows the stability robustness of the controller.

A simple automatic calibration system was implemented for the accelerometers using linear regression. This algorithm attempted to identify the linear parameters to convert volts to m/s^2. These parameters changed with temperature and other factors. Each accelerometer had a slightly different dynamic response. These responses resulted in a variable bias in the sensed relative acceleration. For the joint specific method, this bias was corrected in steady state by the bias filter but did affect the calculated measurements. Although no relative accelerations were used, the difference in accelerometer dynamic response degraded the system wide method's dynamic response.

VII. Experimental results

Before presenting the experimental results, it should be noted that this control method is proposed as a backup method. The position response using accelerometer feedback cannot be expected to be as accurate as that of a controller utilizing optical encoder or resolver position feedback. The goal of these experiments was to show stable performance with position tracking acceptable for continued operation during failure.

Joint specific technique

Figures 9 and 10 show the position tracking response of the system with both the shoulder pitch and elbow pitch joint position sensors failed. Although the RRC robot also provides velocity feedback, this feedback was also considered failed. The joint position data was collected for comparison. The robot was commanded along a trajectory from [0 -π/2 -π -π/2 0 0 0], the Home position, to [0.5 -1.27 -3.5 -1.27 0 0 0] in 1.5 seconds. The two failure case is considered to show how the calculated results for the shoulder pitch joint affect the elbow pitch calculation.

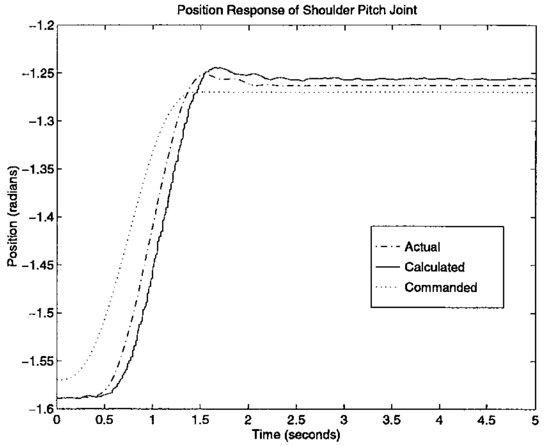

Figure 9: Position response of shoulder pitch joint to trajectory ending at [0.5 -1.27 -3.5 -1.27]

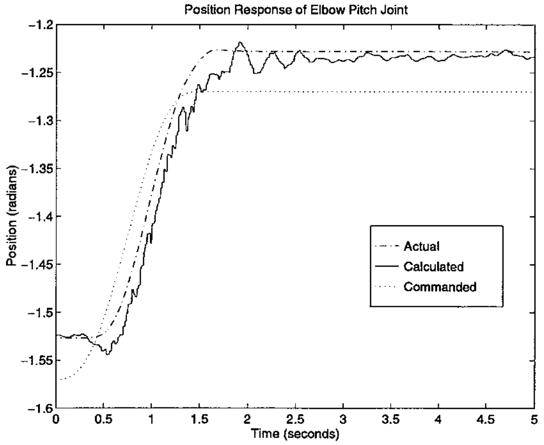

Figure 10: Position response of elbow pitch joint to trajectory ending at [0.5 -1.27 -3.5 -1.27]

As discussed earlier, failures which occur before the joint in question in the kinematic chain will affect the position calculations for the joint. However, joints following that joint will not. This property explains why the wrist joints were not moved in this experiment. The shoulder roll and elbow roll joints were not considered failed in this experiment and were actuated to impart variable dynamic effects on the failed joints making the experiment more interesting.

The shoulder pitch response was better than the elbow pitch response because it sees fewer dynamic effects than the elbow pitch joint. The bias in the steady state response of the elbow pitch joint is due to low position gains. It should be noted that although the actual position did not track the commanded in steady state, the calculated position did track the actual. The damping of the virtual passive controller contributes to the smooth position tracking response.

Figures 11 and 12 show the response of moving from Home to [1 -1 -4 -1] in 1.5 seconds.

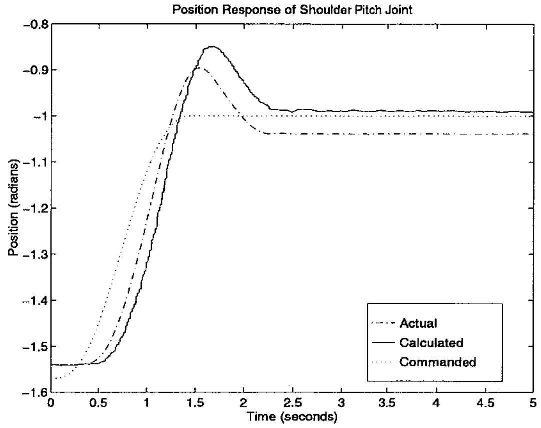

Figure 11: Position response of shoulder pitch joint to trajectory ending at [1 -1 -4 -1]

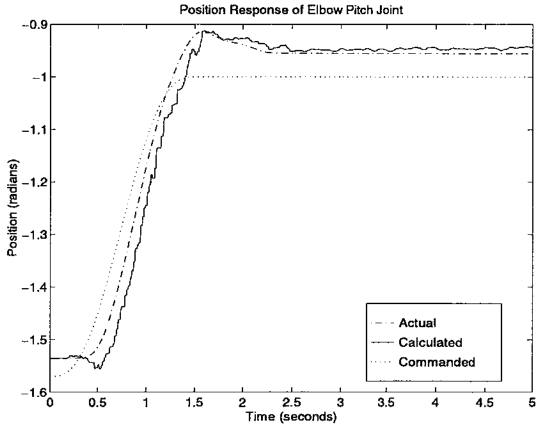

Figure 12: Position response of elbow pitch joint to trajectory ending at [1 -1 -4 -1]

This trajectory is faster than the previous trajectory. As a result, the dynamic effects are more intense and it is more difficult for the PD position controller to start and stop the trajectory as commanded. The faster trajectory also shows the effect of the delay between actual and calculated position. This delay is caused by filter and internal communication delays. This combination of effects results in a substantial overshoot in the shoulder pitch response. This overshoot can be reduced at the cost of poorer tracking of slower trajectories. This is function of PD controllers in general but is worsened by the position calculation delay. The steady state error between calculated and actual position is due to calibration error. The automatic calibration algorithm was optimized for positions significantly far away from the final configuration in this trajectory.

System wide technique

The most interesting failure considered using the system wide method involved three simultaneous position sensor failures. Several sets of failed joints were successfully controlled with the same accelerometer configuration. The representative position response is shown in Figs. 13-15. The commanded trajectory ran from [0 -π/2 -π -π/2 0 0 0] to [0.5 -1.27 -3.5 -1.27 0.3 -0.5 -0.5] in 1.5 seconds. The position sensors for the elbow pitch, wrist pitch, and toolplate roll joints were failed in this case. Different failure sets produced similar results. Lowering the number of simultaneous failures improved results.

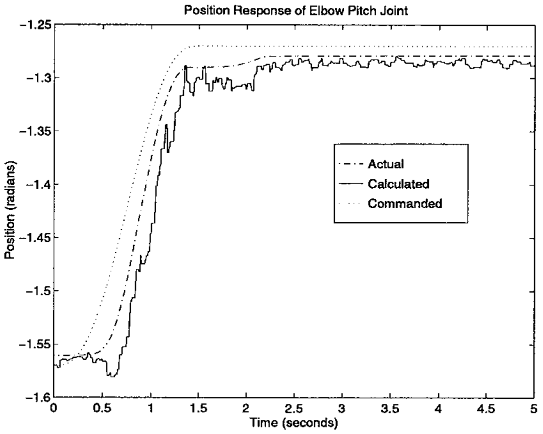

Figure 13: Position response of elbow pitch joint

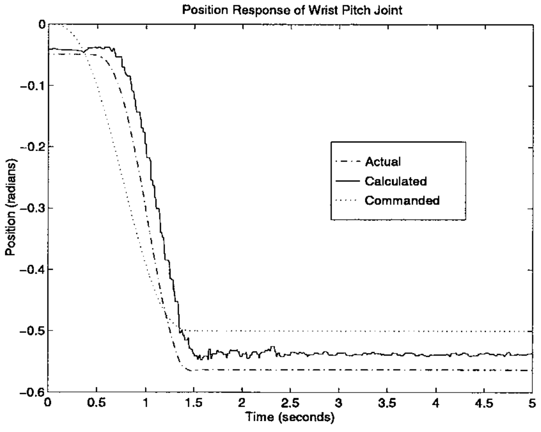

Figure 14: Position response of wrist pitch joint

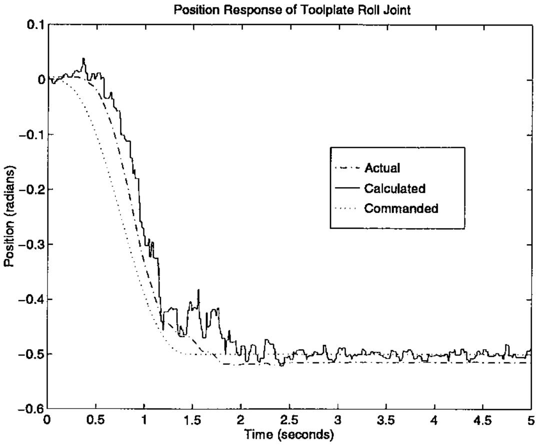

Figure 15: Position response of toolplate roll joint

Similarly to the joint specific method, as the speed of the trajectories is increased, the dynamic effects are more intense and it is more difficult for the PD position controller to start and stop the trajectory as commanded. Figure 16 shows the response of the elbow pitch joint along a trajectory faster than that shown in Fig. 13.

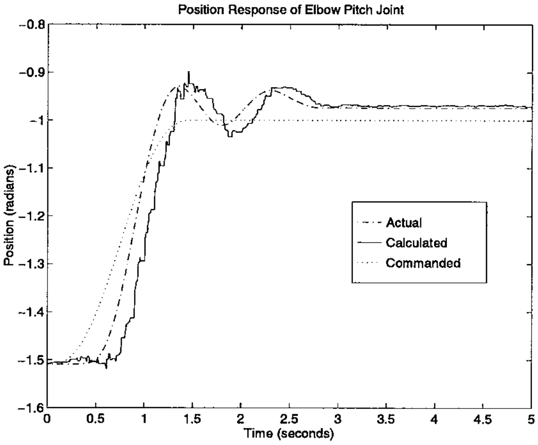

Figure 16: Position response of elbow pitch joint

General notes

The trajectories shown are faster than would reasonably be expected for a robot with failed primary systems. They are shown to illustrate the stability and tracking performance in extreme conditions. For real use, the position gains would be tuned for better tracking at low speeds and the maximum velocity commands limited. As part of a system with a slow end-effector feedback capability to remove small position bias, such as a teleoperated system, the ability to stably continue operation during failure would be more advantageous than improved high-speed operation.

The main problem encountered in this experiment with the accelerometer based position determination method was the signal to noise ratio in certain robot and sensor configurations. In some configurations, the accelerometer axes required to solve for position, not velocity or acceleration, were nearly parallel to gravity. In these configurations, the signal to noise ratio is low and the resulting position calculation suffers. This problem could be solved by choosing trajectories to avoid these known configurations or by better sensor placement.

Another advantage of the proposed control system is that little reconfiguration is required to transition from working to failed states. The performance of the elbow pitch joint with encoder feedback along the same trajectory as shown in Fig. 16 is shown in Fig. 17. This performance is similar to the RRC controller's position control mode. This performance is obtained using the low position gains for failed operation. With these gains, to change from failed to unfailed operation, the position feedback source for the position controller is the only change necessary. The virtual passive torque controller will damp out control error due to a small discontinuity in position feedback during switching. If additional time is required to initialize the fault tolerant components, simply zero position controller output and the virtual passive controller will attempt to slow the joint to a stop.

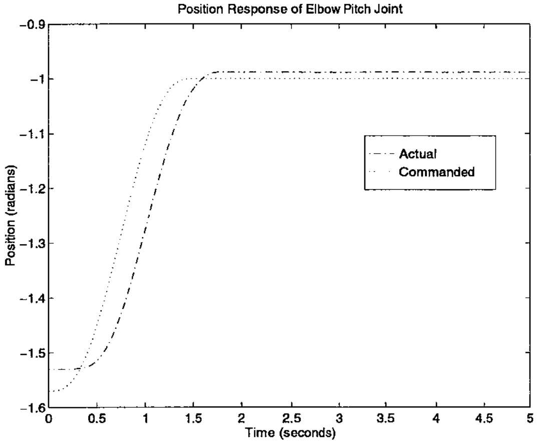

Figure 17: Position response of elbow pitch joint using passive controller with low gains

If the designer wishes to exploit the performance enhancing capabilities of the control system, simply raising the position gains leads to the response in Fig. 18. With these gains the position controller can overcome the residual stiffness and damping on the virtual passive controller which helps during failed conditions but reduces position tracking performance. The system still benefits from sensing static and dynamic loads. As a result, the system will maintain similar tracking in all loading conditions before actuator saturation. To revert to failed operation, the position gains must be lowered. However, the virtual passive torque controller will mask most of the discontinuity of the change. The virtual passive controller gains were constant for all experiments in this paper.

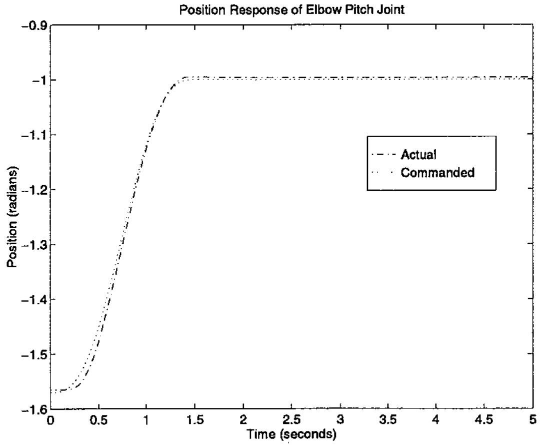

Figure 18: Position response of elbow pitch joint using passive controller with high gains

VIII. Comparison on position determination methods

Both position determination methods summarized here have specific strengths and weaknesses. The joint specific method only provides fault tolerance to the joint which drives the link to which the accelerometers are attached. If only one or two joints in the robot systems are required to be fault tolerant, then the joint specific method is ideal. It will provide position feedback in a predictable, computationally simple manner.

The cumulative errors inherent to both methods make continued operation with more than three simultaneous failures impractical. The main advantage to using the system wide methods is not its theoretical ability to handle more simultaneous failures with fewer accelerometers than the joint specific method. The main advantage of the system wide method is that it can provide fault tolerance to many sets of different joint failures. The same set of accelerometers can provide fault tolerance an elbow joint, a wrist joint, or both. This flexibility allows the system designer to provide system wide fault tolerance with a reasonable number of accelerometers.

The flexibility of the system wide methods comes at a significant cost. The optimization method required by the system wide method is computationally intensive. In addition, the solvability of the system wide method depends on the joints that are failed, the trajectory, and the placement of the accelerometers. The solvability of the failed system along a trajectory must be checked before the robot command is given or the robot can become unstable. A method involving checking the singular values of the determinant of the gradient Q is described in [10].

A more complete discussion of the torque controller and both position determination methods can be found in [14].

IX. Conclusions

This paper has presented a method for continuing operation during the failure of a joint position sensor. The method can be retrofitted into current advanced robot designs, is internal to the robot, and is suitable for long term operation in unstructured environments. The proposed system includes a virtual passive torque controller which does not rely on position feedback for stability and one of two accelerometer based position determination method which do not use integration to obtain position. Controller designs using both position determination methods were described. The proposed controllers were implemented on a robot arm and experimental results presented. The results show stable performance with reasonable tracking during multiple simultaneous joint failures even along trajectories which are more difficult than would be expected of a robot with failed primary systems.

X. Bibliography

[1] Ting, Y., Tosunoglu, S., and Fernandez, B., "Control Algorithms for Fault-Tolerant Robots", 1994 IEEE International Conference on Robotics and Automation, vol. 1, pp. 910-915, 1994.

[2] Visinsky, M.L., Walker, I.D., and Cavallaro, J.R., "Layered Dynamic Fault Detection and Tolerance for Robots", 1993 IEEE International Conference on Robotics and Automation, pp. 180-187, 1993.

[3] Chladek, J.T., "Fault Tolerance for Space Based Manipulator Mechanisms and Control System", First International Symposium On Measurement and Control in Robotics, 1990.

[4] Paredis, C.J.J., Au, W.K.F., and Khosla, P.K., "Kinematic Design of Fault Tolerant Manipulators", Computers and Electrical Engineering, vol. 20, no. 3, pp. 211-220, 1994.

[5] Luh, J.Y., Walker, M., and Paul, R.P., "Resolved acceleration control for mechanical manipulators", IEEE Transactions on Automatic Control, vol. 25, pp. 468-474, 1980.

[6] Kosuge, K., Takeuchi, H., and Furita, K., "Motion Control of a Robot Arm Using Joint Torque Sensors", IEEE Transactions on Robotics and Automation, vol. 6, no. 2, pp. 258-263, 1990.

[7] Imura, J., Yokohohji, Y., and Yoshikawa, T., "Robust Control of Robot Manipulators Based on Joint Torque Sensor Information", International Journal of Robotics Research, vol. 13, no. 5, pp. 434-442, 1994.

[8] Aldridge, H.A. and Juang, J-N., "Virtual Passive Controller for Robot Systems Using Joint Torque Sensors", Fourth IASTED Conference on Robotics and Manufacturing, pp. 73-76, 1996.

[9] Aldridge, H.A. and Juang, J-N., "Joint Position Sensor Fault Tolerance in Robot Systems using Cartesian Accelerometers" AIAA Guidance, Navigation, and Control, paper AIAA 96-3898, 1996.

[10] Aldridge, H.A., and Juang, J-N., "System wide joint position sensor fault tolerance in robotics using Cartesian accelerometers", to be published in SPIE Intelligent Systems and Advanced Manufacturing Conference, November 1996.

[11] Juang, J-N., Wu, S., Phan, M., and Longman, R.W., "Passive Dynamic Controllers for Nonlinear Mechanical Systems", AIAA Journal of Guidance, Control, and Dynamics, vol. 16, no. 5, pp. 845-851, 1993.

[12] Gerald, C.F. and Wheatly, P.O., Applied Numerical Analysis, 5th edition, Reading: Addison-Wesley, 1994.

[13] Press, W.H., Flannery, B.P., Teukolsky, S.A., and Vetterling, W.T., Numerical Recipes in C, Cambridge: Cambridge University Press, 1988.

[14] Aldridge, Hal A., Robot Position Sensor Fault Tolerance, Ph.D. dissertation, Carnegie-Mellon University, 1996.

Appendix A: Gradient calculation

Most nonlinear solution methods require a function gradient for rapid convergence. The closed-form solution for the gradient of an n link manipulator becomes more complex quickly as n grows. The gradient could be computed numerically using finite differences[13]. Finite differences are not as accurate as a closed-form solution. The approach taken here is to calculate the gradient recursively. The system equations are designed to be calculated recursively, so the proper application of partial derivatives and the chain rule will lead to the desired outcome. The desired gradient has the form:

$$\frac{\partial Q}{\partial \Omega} = \left[\begin{matrix} \frac{\partial Am_1}{\partial \theta_1} & \cdots & \frac{\partial Am_1}{\partial \theta_n} & \frac{\partial Am_1}{\partial \dot{\theta}_1} & \cdots \\ \vdots & & \vdots & \vdots & \\ \frac{\partial Am_m}{\partial \theta_1} & \cdots & \frac{\partial Am_m}{\partial \theta_n} & \frac{\partial Am_m}{\partial \dot{\theta}_1} & \cdots \end{matrix} \right. \\ \left. \begin{matrix} \frac{\partial Am_1}{\partial \dot{\theta}_n} & \frac{\partial Am_1}{\partial \ddot{\theta}_1} & \cdots & \frac{\partial Am_1}{\partial \ddot{\theta}_n} \\ \vdots & \vdots & & \vdots \\ \frac{\partial Am_m}{\partial \dot{\theta}_n} & \frac{\partial Am_m}{\partial \ddot{\theta}_1} & \cdots & \frac{\partial Am_m}{\partial \ddot{\theta}_n} \end{matrix} \right] \qquad \text{(A1)}$$

For simplicity, the measurement Am_i is always in the i^{th} frame. Now,

$$\frac{\partial Am_{i+1}}{\partial \theta_k} = \frac{\partial \dot{\omega}_{i+1}^{i+1}}{\partial \theta_k} \times pm_{i+1} + \frac{\partial \omega_{i+1}^{i+1}}{\partial \theta_k} \times \omega_{i+1}^{i+1} \times pm_{i+1} + \omega_{i+1}^{i+1} \times \frac{\partial \omega_{i+1}^{i+1}}{\partial \theta_k} \times pm_{i+1} + \frac{\partial a_{i+1}^{i+1}}{\partial \theta_k} \qquad (A2)$$

$$\frac{\partial Am_{i+1}}{\partial \dot{\theta}_k} = \frac{\partial \dot{\omega}_{i+1}^{i+1}}{\partial \dot{\theta}_k} \times pm_{i+1} + \frac{\partial \omega_{i+1}^{i+1}}{\partial \dot{\theta}_k} \times \omega_{i+1}^{i+1} \times pm_{i+1} + \omega_{i+1}^{i+1} \times \frac{\partial \omega_{i+1}^{i+1}}{\partial \dot{\theta}_k} \times pm_{i+1} + \frac{\partial a_{i+1}^{i+1}}{\partial \dot{\theta}_k} \qquad (A3)$$

$$\frac{\partial Am_{i+1}}{\partial \ddot{\theta}_k} = \frac{\partial \dot{\omega}_{i+1}^{i+1}}{\partial \ddot{\theta}_k} \times pm_{i+1} + \frac{\partial a_{i+1}^{i+1}}{\partial \ddot{\theta}_k} \qquad (A4)$$

The partial derivative will be zero for all quantities of joint *k* where *k>i+1*. This a consequence of the later joints not affecting the previous links.

Equations (A2),(A3),and (A4) are not recursive themselves, but the equations for the partials they depend on are recursive. The equations require that Eqs. (17) through (19) be calculated for all *n* links to determine the angular velocities, angular accelerations, and Cartesian accelerations. The partial derivatives can then be calculated in the following manner:

$$\frac{\partial \omega_{i+1}^{i+1}}{\partial \theta_k} = \begin{cases} R_i^{i+1} \dfrac{\partial \omega_i^i}{\partial \theta_k} & \mathrm{k < i+1} \\ \dot{R}_k^{k+1} \omega_k^k & \mathrm{k = i+1} \\ \bar{0} & \mathrm{k > i+1} \end{cases} \qquad (A5)$$

$$\frac{\partial \omega_{i+1}^{i+1}}{\partial \dot{\theta}_k} = \begin{cases} R_i^{i+1} \dfrac{\partial \omega_i^i}{\partial \dot{\theta}_k} & \mathrm{k < i+1} \\ \hat{z} & \mathrm{k = i+1} \\ \bar{0} & \mathrm{k > i+1} \end{cases} \qquad (A6)$$

$$\frac{\partial \omega_{i+1}^{i+1}}{\partial \ddot{\theta}_k} = \bar{0} \qquad \forall \mathrm{k} \qquad (A7)$$

$$\frac{\partial \dot{\omega}_{i+1}^{i+1}}{\partial \theta_k} = \begin{cases} R_i^{i+1} \dfrac{\partial \dot{\omega}_i^i}{\partial \theta_k} + R_i^{i+1} \dfrac{\partial \omega_i^i}{\partial \theta_k} \times \dot{\theta}_k \hat{z} & \mathrm{k < i+1} \\ \dot{R}_i^{i+1} \dot{\omega}_i^i + \dot{R}_i^{i+1} \omega_i^i \times \dot{\theta}_k \hat{z} & \mathrm{k = i+1} \\ \bar{0} & \mathrm{k > i+1} \end{cases} \qquad (A8)$$

$$\frac{\partial \dot{\omega}_{i+1}^{i+1}}{\partial \dot{\theta}_k} = \begin{cases} R_i^{i+1} \dfrac{\partial \dot{\omega}_i^i}{\partial \dot{\theta}_k} + R_i^{i+1} \dfrac{\partial \omega_i^i}{\partial \dot{\theta}_k} \times \dot{\theta}_k \hat{z} & \mathrm{k < i+1} \\ R_i^{i+1} \omega_i^i \times \hat{z} & \mathrm{k = i+1} \\ \bar{0} & \mathrm{k > i+1} \end{cases} \qquad (A9)$$

$$\frac{\partial \dot{\omega}_{i+1}^{i+1}}{\partial \ddot{\theta}_k} = \begin{cases} R_i^{i+1} \dfrac{\partial \dot{\omega}_i^i}{\partial \ddot{\theta}_k} & \mathrm{k < i+1} \\ \hat{z} & \mathrm{k = i+1} \\ \bar{0} & \mathrm{k > i+1} \end{cases} \qquad (A10)$$

$$\frac{\partial a_{i+1}^{i+1}}{\partial \theta_k} = \begin{cases} R_i^{i+1} \begin{pmatrix} \dfrac{\partial \dot{\omega}_i^i}{\partial \theta_k} \times l_i + \dfrac{\partial \omega_i^i}{\partial \theta_k} \times \omega_i^i \times l_i + \\ \omega_i^i \times \dfrac{\partial \omega_i^i}{\partial \theta_k} \times l_i + \dfrac{\partial a_i^i}{\partial \theta_k} \end{pmatrix} & \mathrm{k < i+1} \\ \dot{R}_i^{i+1} \left(\dot{\omega}_i^i \times l_i + \omega_i^i \times \omega_i^i \times l_i + a_i^i \right) & \mathrm{k = i+1} \\ \bar{0} & \mathrm{k > i+1} \end{cases} \qquad (A11)$$

$$\frac{\partial a_{i+1}^{i+1}}{\partial \dot{\theta}_k} = \begin{cases} R_i^{i+1} \begin{pmatrix} \dfrac{\partial \dot{\omega}_i^i}{\partial \dot{\theta}_k} \times l_i + \dfrac{\partial \omega_i^i}{\partial \dot{\theta}_k} \times \omega_i^i \times l_i + \\ \omega_i^i \times \dfrac{\partial \omega_i^i}{\partial \dot{\theta}_k} \times l_i + \dfrac{\partial a_i^i}{\partial \dot{\theta}_k} \end{pmatrix} & \mathrm{k < i+1} \\ \bar{0} & \mathrm{k \geq i+1} \end{cases} \qquad (A12)$$

$$\frac{\partial a_{i+1}^{i+1}}{\partial \ddot{\theta}_k} = \begin{cases} R_i^{i+1} \left(\dfrac{\partial \dot{\omega}_i^i}{\partial \dot{\theta}_k} \times l_i + \dfrac{\partial a_i^i}{\partial \dot{\theta}_k} \right) & \mathrm{k < i+1} \\ \bar{0} & \mathrm{k \geq i+1} \end{cases} \qquad (A13)$$

where,

$$\dot{R}_i^{i+1} = \begin{bmatrix} -\sin\theta_{i+1} & \cos\theta_{i+1}\cos\alpha_i & \cos\theta_{i+1}\sin\alpha_i \\ -\cos\theta_{i+1} & -\sin\theta_{i+1}\cos\alpha_i & -\sin\theta_{i+1}\sin\alpha_i \\ 0 & 0 & 0 \end{bmatrix} \qquad (A14)$$

is the derivative of R_i^{i+1} with respect to θ_{i+1}.

Using Eqs. (A1) through (A14), the elements of the gradient can be calculated. These equations have been verified against numerically calculated gradients of *Q*.

REPORT DOCUMENTATION PAGE	*Form Approved* *OMB No. 0704-0188*

1. AGENCY USE ONLY (*Leave blank*)	2. REPORT DATE March 1997	3. REPORT TYPE AND DATES COVERED Technical Memorandum

4. TITLE AND SUBTITLE Experimental Robot Position Sensor Fault Tolerance Using Accelerometers and Joint Torque Sensors	5. FUNDING NUMBERS 519-30-21-04
6. AUTHOR(S) Hal A. Aldridge Jer-Nan Juang	
7. PERFORMING ORGANIZATION NAME(S) AND ADDRESS(ES) NASA Langley Research Center Hampton, VA 23681-0001	8. PERFORMING ORGANIZATION REPORT NUMBER
9. SPONSORING / MONITORING AGENCY NAME(S) AND ADDRESS(ES) National Aeronautics and Space Administration Washington, DC 20546-0001	10. SPONSORING / MONITORING AGENCY REPORT NUMBER NASA TM 110335
11. SUPPLEMENTARY NOTES	
12a. DISTRIBUTION / AVAILABILITY STATEMENT Unclassified-Unlimited Subject Category 31	12b. DISTRIBUTION CODE

13. ABSTRACT *(Maximum 200 words)*

Robot systems in critical applications, such as those in space and nuclear environments, must be able to operate during component failure to complete important tasks. One failure mode that has received little attention is the failure of joint position sensors. Current fault tolerant designs require the addition of directly redundant position sensors which can affect joint design. The proposed method uses joint torque sensors found in most existing advanced robot designs along with easily locatable, lightweight accelerometers to provide a joint position sensor fault recovery mode. This mode uses the torque sensors along with a virtual passive control law for stability and accelerometers for joint position information. Two methods for conversion from Cartesian acceleration to joint position based on robot kinematics, not integration, are presented. The fault tolerant control method was tested on several joints of a laboratory robot. The controllers performed well with noisy, biased data and a model with uncertain parameters.

14. SUBJECT TERMS Robot control; Fault tolerance; Torque sensor; Accelerometer	15. NUMBER OF PAGES 14
	16. PRICE CODE AO3

17. SECURITY CLASSIFICATION OF REPORT Unclassified	18. SECURITY CLASSIFICATION OF THIS PAGE Unclassified	19. SECURITY CLASSIFICATION OF ABSTRACT Unclassified	20. LIMITATION OF ABSTRACT